Determining whether or not a given string is a palindrome

(An Application of a stack and a queue)

A string of characters is a palindrome if and only if it reads the same forward and backward. Examples: eye, abba, madam, atoyota, malayalam.

A recursive definition follows.

(1) The empty string is a palindrome.
(2) A string consisting of a single character is a palindrome.
(3) If $w$ is a palindrome and $a$ is a letter in the alphabet, then $awa$ is a palindrome.
(4) A string of characters is a palindrome if and only if its being so follows from finitely many applications of rules (1) through (3) above.

Method of attack

- Read the input string one character at a time.
- Push a copy of the character on to the stack. At the same time, add a copy to (the back of the) queue.
- When the line has been entirely read, the program repeatedly compares the top of element of the stack and the front element of the queue; in case of a match, it pops the stack and (simultaneously) de-queues the queue.
- If ever there is a mismatch between the top element of the stack and the front element of the queue, the program declares that the input string is not a palindrome, and halts.
- If the stack (or equally correctly, the queue) gets empty, the program declares that the given string is a palindrome, and halts.

Correctness follows from the fact that a queue preserves the order of items recorded in it whereas a stack reverses that order.

Because of the high importance of a stack and a queue as data structures in computer science, C++ includes their implementation in the standard template library.

Library headers <stack> and <queue> place the definitions of a stack and a queue, respectively, in the std namespace. Important member functions of the two classes are as follows.
Here is a sample dialog.

Please enter a string of characters: abba

The given string is a palindrome.

Want to examine another string? (y/n): y
Please enter a string of characters: 11223311

The given string is not a palindrome, since the symbol at position 3 from the left is different from the symbol at position 3 from the right.

Want to examine another string? (y/n): n

Bye!

Note: User inputs in the preceding dialog are in blue.

A program appears below.
Program to determine whether a given string of characters is a palindrome.

```cpp
#include <iostream>
#include <stack> // stack template exists in the system library.
#include <queue> // queue template exists in the system library.
using namespace std;

int main()
{
    char ch; // Variable used to hold an input character.
    char ans; // Variable used in the dialog: Want to examine another string? (y/n)
    bool good;
    int i;
    do // Beginning of the do-while loop.
    {
        stack<char> s; queue<char> q;
        // Declaring s and q here ensures that the stack s and queue q
        // are necessarily empty at the beginning of each iteration.
        cout << "Please enter a string of characters: ";
        cin.get(ch);
        while(ch != 'n') // Read the input string one character at a time.
        {
            s.push(ch); q.push(ch);
            // A copy of ch goes at the top of the stack. At the same
            // time, a copy of ch is added at the end of the queue.
            cin.get(ch);
        } // end of while
        good = true; i = 1;
        while(!s.empty()) // Repeatedly compare the top element of the stack
        {
            // and the front element of the queue
            if (s.top() == q.front())
            {
                s.pop(); q.pop(); i++;
            }
            else
            {
                good = false; break;
            }
        } // end of while
        if(good)
        {
            cout << endl << "It is a palindrome.;"
        }
        else
        {
            cout << endl << "It is not a palindrome, since"
            << endl
            << "the symbol at position " << i
            << " from the left is different from " << endl
            << "the symbol at position " << i
            << " from the right."
        }
        cout << endl << "Want to examine another string? (y/n): ";
        cin >> ans;
        cin.ignore(100, 'n'); // Ignore the newline character.
        while(ans != 'n' && ans != 'N' && ans != 'Y' && ans != 'Y')
        {
            // Force the user to input n or N or y or Y.
            cout << "Please enter n or N or y or Y: ";
            cin >> ans;
            cin.ignore(100, 'n'); // Ignore the newline character.
        } // end of while
    } while (ans == 'y' || ans == 'Y'); // end of do-while.
    cout << endl << "Bye!");
    return 0;
} // main
```